1973

Prefix Codes, Trees and Automata

Jean-Louis Lassez

Report Number:
73-106

PREFIX CODES, TREES, AND AUTOMATA.

(To appear in INFORMATION SCIENCES.)

Jean-Louis Lassez
Department of Computer Science
Purdue University
Lafayette, Indiana 47907

CSD TR 106

Prefix codes, trees, and Automata.

(To appear in INFORMATION SCIENCES.)

Jean-Louis Lassez

Department of Computer Science

Purdue University

Lafayette, Indiana 47907

This work was supported by grant No. GJ-980 from the N.S.F. and CNR contract No. R-17-02-417-0-A.

Prefix codes, trees, and [illegible]

(To appear in INFOR[illegible])

Abstract

We investigate the relations between the tree representation of a prefix code C, its patterns, the minimal automaton $\mathcal{G}$ which recognizes C^* and the homomorphic images of the underlying algebra of $\mathcal{G}$. These relations provide us with an appealing formulation which allows us to state (and eventually solve) some problems concerning the structure of prefix codes and therefore the structure of trees.

Introduction.

In this paper we study the structure of prefix codes and the structure of their tree representation. Besides the introduction of the notion of pattern and the derived formulation, Part I will be an exposition of known results (cf classical works of Schützenberger, Nivat, Perrot). In Part II we study an important family: the overlapping codes, and provide a new demonstration of a basic theorem due to Perrin and Perrot. In Part III we bring partial answers to the problems raised in Part I and derive some applications.

The reader has to be aware that the intuitive use of the geometrical representation of how the pattern is distributed in the tree provides a frame for many demonstrations which are otherwise heavily technical. We assume known the basic results on regular languages and finite automata.

Notations.

X will denote a finite set of cardinality strictly superior to one, the alphabet. All the examples, for sake of simplicity, will be given using a binary alphabet. The reader will be easily convinced that the assumption of finiteness for X may be removed for many results.

X^* will denote the free monoid over X, and "e" the null word.
X^+ will denote the free semigroup over X: $X^* = X^+ + e$.

If L is a language of X^*, the right congruence $=$ induced by L is defined as follows on the words of X^*:

$$u = v \ (L) \quad \text{iff} \quad \forall h \in X^* \quad uh \in L \Leftrightarrow vh \in L.$$

A prefix exhaustive code is a subset C of X^* such that no word of C is a strict prefix of another word of C and every word of X^* is a prefix of a word of C or admits a word of C as a prefix, in equivalent terms:

$$\forall h \in X^* \quad C^*h \cap C^* \neq \emptyset \Rightarrow h \in C^*.$$

$$\forall h \in X^* \quad hX^* \cap C^* \neq \emptyset$$

These properties will be used constantly and we will not refer systematically to them. Furthermore, as it is done usually we will use the word "code" instead of "prefix exhaustive code".

$C^{\downarrow} = X^* - CX^+$ will denote the set of the prefixes of the words of C.

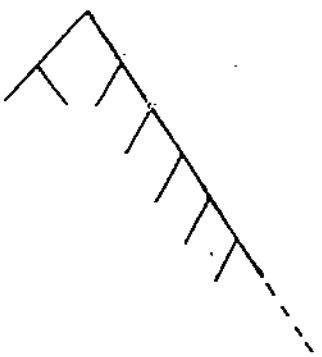
Part I: Generalities.

Codes are characterized by a tree representation such that the

same number of vertices are issued from each non terminal node. The following examples should be clear enough so we do not need to give an unnecessarily heavy formulation:

The / vertices represent the letter x and \ the letter y.

represents the
finite code: $\{x, yxx, yxy, yy\}$

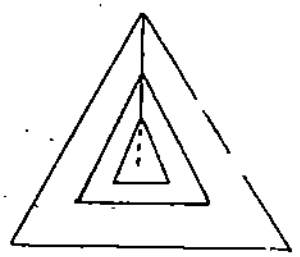


represents the
regular code: $\{xx, xy, yy^*x\}$

represents the non-regular code:

$$C = x + \sum_{n=1}^{\infty} \left(y^n \left(\sum_{p=1}^{p=n} x^p y \right) + y^n x^{n+1} \right)$$

Some properties of C may be better seen on the tree representation that on the complex formula. In particular, one can see that there exist words h in $C^!$ such that: $hC \subset C^!$. Such a code will be called a "matriochka" code, because in some sense it is contained in itself.

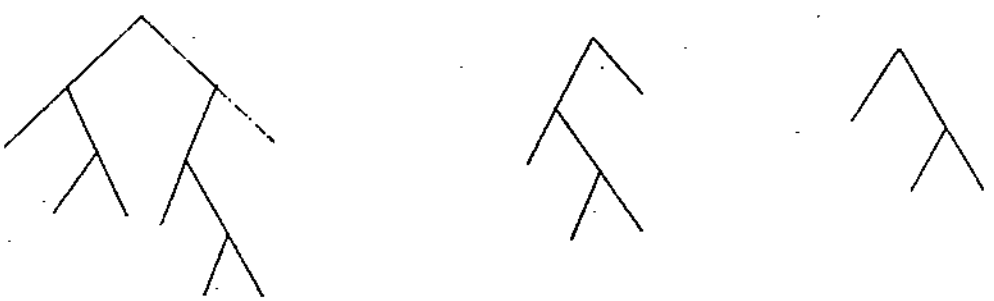


In the remaining part we will use the same notation for the tree and the code, when no confusion will be possible.

Patterns:

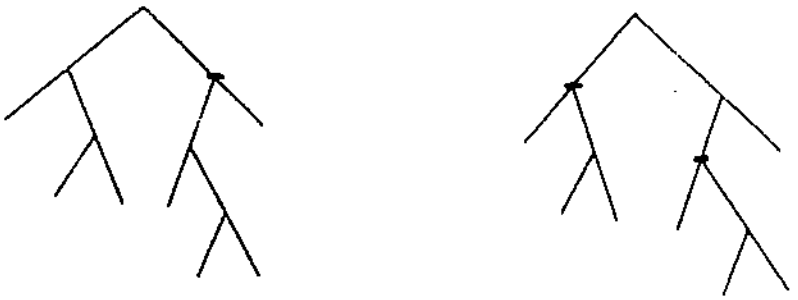
In fact the tree represents $C^!$ and not only C and so if a code is "composed" of other codes it is easily seen on the tree:

Example:



A B C

We see that A may be built using B or C:



We will express this situation by saying that the trees B and C are "patterns" of the tree A.

There are various ways to formalize this notion, we will give a "constructive" one which will be appropriate to further demonstrations.

Let A and B be two codes and define:

$$B_o = \{e\} \quad , \quad B_{i+1} = (B_i - A)B$$

Definition: B is a pattern of A if and only if for every integer i $B_i \subset A^1$.

This is a different way to define the notions of supercoding (Schutzenberger [12]) and composition (Nivat [4]). The following theorem shows the utility of this definition:

Theorem I.1: $A^* \subset B^*$ iff B is a pattern of A.

Proof: Let us suppose that B is not a pattern of A, therefore we have a first integer i such that $B_i \neq A^1$. So we have $b \in B_i$ and $b \notin A^1$ (therefore $b \neq e$ and $i > 0$) and $B_{i-1} \subset A^1$. We know then that $b = da$, with $d \in B_{i-1}$, $d \notin A$ and $a \in B$, therefore $d \in A^1$. Because $b \notin A^1$ and A is exhaustive we have $c \in A$ and c is a strict prefix of b. As $b = da$ and $d \in \{A^1-A\}$ this implies that d is a strict prefix of c. So there exists $u \neq e$ such that $c = du$ and clearly u is a strict prefix of $a \in B$. As B is prefix this implies $u \notin B^*$ and since $d \in B^*$ we have $c \notin B^*$. As $c \in A$ we therefore have $A^* \not\subset B^*$.

Conversely let us suppose that $A^* \not\subset B^*$ it is equivalent to say that there exists $a \in A$ such that $a \notin B^*$. If a is a prefix of a word of B, then $B_1 = B \not\subset A^1$. If a admits a word of B as a prefix, then a may be written as $a = bh$ where $b \in B^*$ and $h \neq e$ is a strict prefix of $c \in B$. Not any prefix of b belongs to A since $a \in A$ and A is prefix, therefore there exists an integer n such that $b \in B_n$, then $bc \in B_{n+1}$ and does not belong to A^1 since $a \in A$ is one of its strict prefixes.

Tree representation and automata.

Let $\mathcal{G} = \langle S, X \rangle$ be an universal algebra where S is a set whose elements are called states and where X is a set of unary operations called the alphabet. The action of the elements of X on S is naturally extended to the elements of X^*. All the algebras considered here will be assumed strongly connected, that is $\forall s, t \in S \; \exists f \in X^*$ such that $sf = t$.

Let $\mathcal{G}_o = \langle S, s_o, X \rangle$ denote the algebra $\mathcal{G}$ in which an element s_o of S has been distinguished, then the set $C^* = \{ f \in X^* \mid s_o f = s_o \}$ is generated by a code (prefix exhaustive) C, as it is clearly verified. We will say that $\mathcal{G}_o$ is an automaton which recognizes C^*.

Conversely, let us suppose that C is a code, there exists at least one strongly connected algebra $\mathcal{A}(C^*) = \langle S, X \rangle$ and a distinguished element s_o in S such that $\langle S, s_o, X \rangle$ recognizes C^*.

Indeed we take S as being the set of classes of the right congruence induced by C^* on the words of X^* and s_o as being the class of the empty word. The words of X^* operate naturally on the set S. $\langle S, s_o, X \rangle$ is then a minimal automaton recognizing C^*. ([1],[10]).

The following properties of $\mathcal{A}(C^*) = \langle S, X \rangle$ and of $\langle S, s_o, X \rangle$ are classical.

Property I2: In every class s there exists at least one element of C^1.

Proof: Let f be an word in a class s. As C is exhaustive we may write $f = ch$ where $c \in C^*$ and $h \in C^1$, it is clear that $f \equiv h(C^*)$ and then h is in the class s.

Property I3: If f and g belong to $X^+ - CX^*$ then $f \equiv g(C^*)$ if and only if $\{h \in X^* \mid fh \in C\} = \{h \in X^* \mid gh \in C\}$.

Proof: If $f \equiv g(C^*)$ and $fh \in C$, then $gh \in C^*$. We have $s_o f = s_o g = s$, $s_o fh = s_o gh = s_o$. If p is a prefix of gh $s_o p = s_o$ implies $p = e$ or $p = gh$ otherwise C would not be a prefix. Therefore $gh \in C$. Conversely let $F = \{h \in X^* \mid fh \in C\}$ and $G = \{h \in X^* \mid gh \in C\}$ and suppose $F = G$. The set of words u such that $fu \in C^*$ is clearly equal to FC^* and the dual for g. Therefore $f \equiv g(C^*)$.

Property I4: For any congruence on $\mathcal{A}(C^*)$ different from the identity, s_o is not alone in its class.

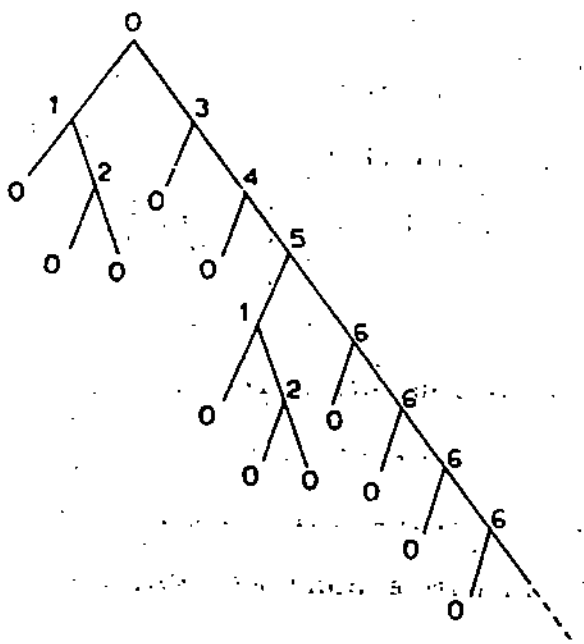
Proof: If there was such a congruence and s_o alone in its class then we would have two distinct states s and t congruent

which would imply $\forall f \in X^*$ $sf = s_o \Leftrightarrow tf = s_o$ in contradiction with the definition of $\mathcal{M}(C^*)$.

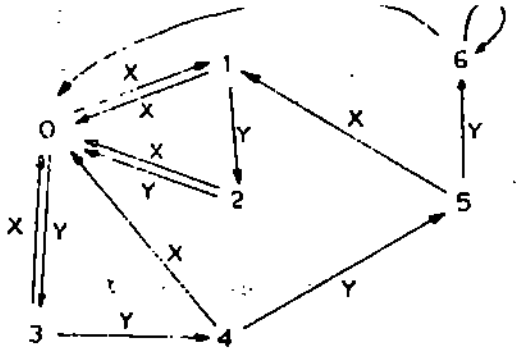
These properties allow us to derive $\mathcal{M}(C^*)$ directly from the tree representation of C: the root and all terminal nodes will be labelled with s_o and only these nodes will be labelled with s_o. We attach the same label to nodes (different from the root and the terminal nodes) which admit the same subtree (Property 13) and this label does not appear elsewhere. The set of labels represents S in $\mathcal{M}(C^*)$ (Property 12).

Example:

$C = \{xx, xyx, xyy, yx, yyx, yyyxx, yyyxyx, yyyxyy, yyyyy^*x\}$



From there we may draw the transition graph:



Patterns and Automata.

Let $\mathcal{A}(C^*) = \langle S, X\rangle$ and $\mathcal{C}_o = \langle s_o, X\rangle$ be a minimal automaton which recognizes C^*. If $\mathcal{D} = \langle T, X\rangle$ is an homomorphic image of $\mathcal{A}(C^*)$, then $\mathcal{D}_o = \langle T, t_o, X\rangle$, where t_o is the class of s_o in the congruence relation induced by the homomorphism, recognizes D^* where D is a code and it is easy to verify that $C^* \subset D^*$. In other terms the homomorphic images of $\mathcal{A}(C^*)$ "extract" patterns of C. It has been shown by Perrin and Perrot [9] that when C is finite there is a bijection between the homomorphic images of $\mathcal{A}(C^*)$ and the patterns of C (this is restated in our terminology).

Definition: We will say that D is an admissible pattern of C if and only if there exists an homomorphic image $\mathcal{D} = \langle S, X\rangle$ of $\mathcal{A}(C^*)$ such that $\mathcal{D}_o = \langle S, t_o, X\rangle$ recognizes D^*. (t_o has been defined above).

Several problems arise from this terminology:

characterize the admissible and the non admissible patterns of a code C.

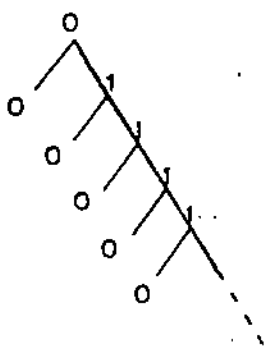
Is it possible to derive the non admissible patterns from the admissible ones? ...

These problems are relevant from the following point of view: the properties of a code C are closely related to its patterns, $\mathcal{M}(C^*)$ is a useful algraic tool to study the properties of C so we have to see how much information on the patterns we may get through $\mathcal{M}(C^*)$ and if we can retrieve the unavailable information by some other means.

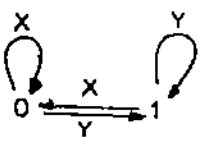
Before we bring partial answers to these problems, we will give a quite simple example:

Code $C = y^*x$

Tree



$\mathcal{M}(C^*)$:

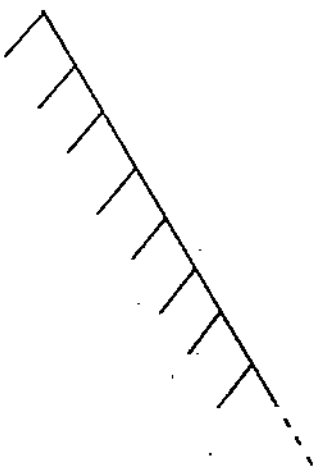


$\mathcal{M}(C^*)$ has only trivial homomorphic images, but C admits infinitely many patterns: $D_n = \sum_{p=0}^{p=n} y^p x + y^n$.

Now it is clear [7] that a pattern D is admissible for C if and only if $D^{\#} \cap C^{1}$ is saturated modulo the right congruence induced by C^{*}. In terms of tree a pattern D of C is not admissible if and only if there exist a node n_1 corresponding to a word of $D^{*} \cap C^{1}$, a node n_2 corresponding to a word which does not belong to D^{*}, both nodes different from the root and the two subtrees below these nodes are equal. (Furthermore these two subtrees admit D as a pattern). Using the previous example:

$C = y^{*}x$ $\qquad$ $D = x + yx + yy$

C



D

The nodes n_1 and n_2 admit C as a subtree.

Part II. Overlapping patterns.

Perrin [6] has studied the following problem in the regular case (restated in our terminology):

Characterization of the codes D such that there exists another code C and D is a non admissible pattern of C. Let us call

non universally admissible such a code. He showed that D is non universally admissible if and only if D is synchronizing in some B^* (id est $\exists\, d \in D^*$ such that $B^* d \subset D^*$) using rather sophisticated properties of the minimal right ideals of the syntactic monoid of D^*. We will study here the general case and find a characterization in terms of tree which will allow us to find a weaker characterization in terms of code. We introduce now a new family of codes derived from the study made at the end of Part I:

Definition: We will say that D is overlapping if and only if one can find C such that

i) D is a pattern of C

ii) There exists $f \in X^+ \cap C^\downarrow$, $f \notin D^*$ and the subtree below f admits D as a pattern.

Another notion is that of strongly non suffix.

Definition: D is said strongly non suffix if and only if $\exists\, f \notin D^*$ such that $\forall d \in D^*\ \exists d' \in D^*$ and $f\, d\, d' \in D^*$.

We have introduced three families of codes, one defined in terms of algebra: the non universally admissible code, one defined in terms of tree: the overlapping codes and one defined in terms of codes: the strongly non suffix codes. The following theorem tells us how they are related

Theorem II.1. The following properties are equivalent

i) D is non universally admissible

ii) D is strongly non suffix

iii) D is overlapping

Proof.

i) → ii). There exists a code C which admits D as a pattern and $\exists f \in D^*$, $\exists d_1 \in D^*$ such that $f \equiv d_1(C^*)$. Let d be any word in D^*, $f\,d \equiv d_1 d\,(C^*)$ as C is exhaustive $\exists h$ such that $f\,d\,h \in D^*$ therefore $d_1 d h \in C^* \subset D^*$ and as D is prefix, $h \in D^*$.

ii) → iii) $\exists f \notin D^*$ such that $\forall d \in D^*$ $\exists d' \in D^*$ and $f\,d\,d' \in D^*$. It is seen easily that we may consider $f \in D^1$, define $E = D^* \cap fD^*$, $F = E - ED^+$ and $C = \{D - fX^*\} \cup F$. If C was not a prefix set there would exist two words fa and fb and a word $h \neq e$ such that $f\,a\,h = fb \in C$ (the two other cases are trivial), but then $ah = b$ and as a and b belong to D^*, h belongs to D^* and therefore $fb \notin C$. We have to show that C is exhaustive. Let us consider $h \in X^*$, if h is a prefix of a word of $\{D - fX^*\} \cup \{f\}$ or admits a word of $\{D - fX^*\}$ as a prefix, the problem is solved. So we may suppose $h = fh'$. As D is exhaustive $\exists h''$ such that $h'h'' = d \in D^*$ and then $hh'' = fh'h'' = fd$. As D is strongly non suffix $\exists d' \in D^*$ such that $fdd' \in D^*$ and so the word

Applications.

Let D be a code with a pattern C such that C is partitioned into $C = C_o + C_1$ and $\{B_i^{-}D\}C_1 \subset D$ with $B_o = \{e\}$ and $B_i = \{B_{i-1}^{-}D\}C$. Such a code will be called a pseudo periodic code with pseudo period C. One can see easily, from the demonstration of the previous theorem that D is a basis of $C_o^*C_1$ even though C is not necessarily the principal pattern of $C_o^*C_1$. We recall that a code A is said synchronizing [12] iff there exists a word $a \in A^*$ such that $X^*a \subset A^*$. It is clear that A is synchronizing if and only if its bases are synchronizing therefore we may state:

Proposition III.6. A pseudo-periodic code is synchronizing if and only if its pseudo-period is synchronizing.

Now if we take a finite tree $C = C_o + C_1$ of cardinality n, if C_1 has a cardinality equal to $n - 1$ all the pseudo-periodic trees we will build will be finite periodic trees or equal to $C_o^*C_1$. But if C_1 has a cardinality strictly inferior to $n - 1$ then we may build a basis of $C_o^*C_1$, regular code, which will not be regular, as shown on the example at the end of the paper. So we may state:

Proposition III.7. A regular code may have non regular patterns.

In the next theorem, which gives a characterization of the bases, $A = C_o^* C_1$ will be a systematic prefix code and $C = C_o + C_1$ will be its principal pattern. If D admits C as a pattern, we will define $B_o = \{e\}$, $B_i = \{B_{i-1} - D\}C$, we recall that D being a pattern of C we have $B_i \subset C^1$ for every i.

Theorem III.5. Let D be a code distinct from A, then: D is a basis of A if and only if C is a pattern of D such that $\forall i \geq 0 \ \{B_i - D\}C_1 \subset D$.

Proof: Let D be a basis of A, Proposition III.2 tells us that C is a pattern of D, the definition of B_i and the fact that C is a pattern of D imply for every i $\{B_i - D\} \subset D^1$. Furthermore if $\{B_i - D\} \cap C^* C_1 C^* \neq \emptyset$ one would find $a \in A$ and $b \in \{B_i - D\}$ such that $b \in aX^*$ and as $a \in D^+$, $b \in D^+X^*$ which is impossible. Therefore $\{B_i - D\} \subset C_o^*$ and $\{B_i - D\}C_1 \subset A \subset D^+$ and as $\{B_i - D\}C_1 \subset D^1$, we have $\{B_i - D\}C_1 \subset D$.

Conversely let us define $D_1 = \bigcup_i \{B_i - D\}C_1$. As $\{B_o - D\}C_1 = C_1 \neq \emptyset$, $D_1 \neq \emptyset$. Furthermore it is clear that $\{B_i - D\} \subset C_o^*$ otherwise D would not be prefix. Now $D = D_1$ would imply $D = D_1 \subset C_o^* C_1 = A$ and then $D = A$, therefore $D_o = D - D_1 \neq \emptyset$. $D_o \subset C_o^*$ since D is prefix and D_o is disjoint from D_1. Therefore we have a partition of D, $D = D_o + D_1$, and as $D_o^* D_1 \subset C_o^* C_1 = A$. $\Rightarrow$ $D_o^* D_1 = A$

Example: Let $C = x + yx + yy$, $A = (yx)^*(x+yy)$ the bases of A are perodic trees with period C: $D = \sum_{k=0}^{k=n} \left[(yx)^k (x+yy) \right] + (yx)^{n+1}$

C

D

However the structure of the bases of systematic prefix codes may be more intricate, in particular the proposition below shows that they may be themselves systematic prefix codes:

Proposition III.4. If $A = C_o^* C_1$ and if C_o may be partitioned into $C_o = C_2 + C_3$, then $D = C_2^*(C_1+C_3)$ is a basis of A.

Proof: According to Proposition III.1 A^+ represents the set of words of C^+ whose last word in the decomposition into words of C belongs to C_1, D^+ represents the set of words whose such last word belong to $\{C_1 + C_3\}$, it is then clear that $A^* \subset D^*$ and that D is a basis of A (but C is not necessarily the principal pattern of D even if it is the principal pattern of A).

Our aim here is to study the lattice of patterns D comprised between the principal pattern C of A and A itself, or in other terms, to characterize the structure of the codes D such that $C^* \subset D^* \subset A^*$. In [2] it was shown that among such codes D, the only admissible cases were $D = C$ or $D = A$. So $\mathcal{M}(A^*)$ bears information only for C, the following study will show us how we can derive the structure of D from the structure of C, the first step being:

Theorem III.3. D is a basis of A if and only if $A^* \subset D^* \subset C^*$

Proof: If D is a basis of A, then $A = D_o^* D_1 \subset D^* = (D_o + D_1)^*$ and so $A^* \subset D^*$, now $D^* \subset C^*$ from Proposition III.2. Conversely, if $A^* \subset D^* \subset C^*$, we have $A^* + \{D^* - A^+\} = D^*$ and $\{D^* - A^+\}$ is included in $\{C^* - A^+\}$ which is a submonoid from Proposition III.1, therefore the product of two elements of $\{D^* - A^+\}$ is in D^* and in $\{C^* - A^+\}$. $\{D^* - A^+\}$ is then a submonoid. To show that it is generated by a prefix set, let us suppose d and $dh \in \{D^* - A^+\}$, as D is prefix we know that h is in D^*, if h was in A^+ then dh would be in A^+ which is a left ideal in C^*, therefore ~~(A is prefix and as $A^* \subset D^*$ and A is exhaustive, then D is exhaustive.)~~ We may then apply Proposition III.1 to end the proof.

$D^* - A^+$ is generated by a prefix code, exhaustive since $A^* \{D^* - A^+\} \subset D^+ - A^+$ and A is exhaustive.

Part III. Bases of systematic prefix codes.

If we consider an infinite regular tree and its labelling we see that if we follow an infinite branch on the tree, necessarily, as the labelling is finite, we will meet two nodes identically labelled. Therefore these two nodes admit the same subtree which may be expressed as $D_0^*D_1$ where D_0, D_1 form a partition of a code $D = D_0 + D_1$, called a basis of $A = D_0^*D_1$. This type of codes, called systematic prefix codes has been introduced by Neumann [5] and plays an important role in Information Theory. Perrin pointed out that they were related to the problem discussed in Part I [6], [7] and derived properties of codes which do not belong to this family. In [2] we studied their structure and derived the two following results that will be used later on:

Proposition III.1. A is a systematic prefix code if and only if A is a code such that there exists a code D, pattern of A and $D^* - A^+$ is a free submonoid. Then there exists a partition of $D = D_0 + D_1$ such that $D_0^*D_1 = A$ and $D_1^*D_0$ generates $D^* - A^+$.

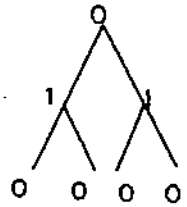
Proposition III.2. Among the bases of a systematic prefix code A there exists one, called the principal pattern of A which is a pattern of all the bases of A.

Lemma II.3: If a code D is overlapping in a code C, then C is infinite.

Proof: If D overlaps in C then $\exists f \notin D^*$, $f \in C^!$ and the subtree below f admits D as a pattern; therefore $fD \subset C^!$. Let $f = d_1 h$ with $d \in D_i$ and $h \in D^!$. If $hD \subset D^!$ (the matriochka case, as we have seen in an example in the introduction) then D is infinite and so is C. If $hD \not\subset D^!$ then $\exists k$ such that $dhk \in D_{i+1}$ since D is exhaustive and $dhk \notin C$ otherwise we would be in the preceding case. Therefore $D_{i+2} = \{D_{i+1} - C\}D$ is not empty. Now if $fkD \subset fD^!$ we are in the matriochka case, if $fkD \not\subset D^!$ then $\exists k'$ such that $fkk' \in C^!$, $fkk' \notin D^*$, $fkk' \in fD$ and so fkk' has a subtree which admits D as a pattern, and so we may reiterate. Therefore C is infinite because D is infinite or $C^!$ admits words of unbounded length.

A direct consequence of this lemma is that if C is finite all its patterns do not overlap in C and therefore are all admissible. We know how to assign a pattern of C to any homomorphic image of $\mathcal{A}(C^*)$, if C is finite the application is injective since any homomorphic image leads to a minimal automaton, and is surjective since every pattern is admissible.

But we see that $\mathcal{D}$ is not isomorphic to $\mathcal{A}(D^*)$ which is



This is due to the fact that no word of $\bigcup_{i=0}^{\infty} D_i x$ has the same subtree in C as a word of $\bigcup_{i=0}^{\infty} D_i y$, and x and y have the subtree in C. Therefore the corresponding labels will form two distinct classes in the congruence. So if the homomorphic image of $\mathcal{A}(C^*)$, $\mathcal{D}$ is not isomorphic to $\mathcal{A}(D^*)$ there exist two words f and y in $D' - D^*$ which have the same subtree in D and such that no word of $\bigcup_{i=0}^{\infty} D_i f$, has the same subtree in C as a word of $\bigcup_{i=0}^{\infty} D_i y$. We see in particular that D_i may be empty and that C is infinite.

Therefore if C is finite any homomorphic image $\mathcal{D} = \langle T, X \rangle$ of $\mathcal{A}(C^*)$ will be such that $\mathcal{D}_o = \langle T, t_o, X \rangle$ is a minimal automaton which recognizes D^*.

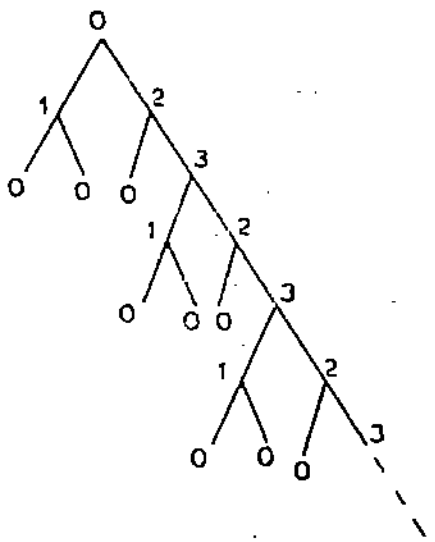
Now, the basic theorem of Perrin and Perrot [9], stating that if C is finite there is a bijection between the patterns of C and the homomorphic images of $\mathcal{A}(C^*)$ will be a consequence of the following lemma and the preceding remark.

Tree representation as an intuitive tool.

Let D be a pattern of C, then by definition, we know that $D_i = \{D_{i=1} - c\}D \subset C^1$.

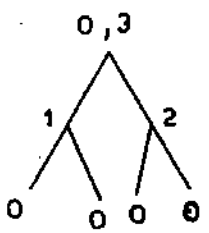
If D is an admissible pattern of C we obtain the homomorphic image of $\mathscr{D}$ of $\mathscr{A}(C^*)$ as shown in the examples:

C:

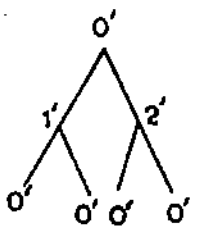


D:

We break C according to D, and we pile up:



We have then a congruence (0,3) (2) (1) on (0,1,2,3) from which we build $\mathscr{D}$



Proof: If c_1 and c_2 belong to C^*, their product belongs to C^* since for every $d \in D^*$ $\exists d' \in D^*$ such that $c_2dd' \in D^*$, but then $\exists d'' \in D^*$ such that $c_1c_2dd'd'' \in D^*$. We have to show that C^* is generated by C, prefix, exhaustive. Let c and ch belong to C^*, does h belong to C^*? $\forall d \in D^* \exists d' \in D^*$ such that $chdd' \in D^*$. As D is exhaustive we may find f such that $hdd'f \in D^*$ and so $\exists d'' \in D^*$ such that $chdd'fd'' \in D^*$. As $chdd' \in D^*$ and D is prefix, we have $fd'' \in D^*$ and so $hdd'fd'' \in D^*$. Therefore, for each d in D^* we have found $b = d'fd'' \in D^*$ such that $hdb \in D^*$. So $h \in C^*$ and C is prefix. Since $D^* \subset C^*$, C is exhaustive and then C is a pattern of D. If $c \in C^*$ and $f \in X^*$ are such that $c \equiv f\ (D^*)$ then for every d in D^* $cd \equiv fd(D^*)$. As $\exists d' \in D^*$ such that $cdd' \in D^*$ we have $cdd' \equiv fdd' \equiv e(D^*)$, therefore $fdd' \in D^*$ and $f \in C^*$. C^* is then saturated modulo D^*.

$hh''d \in E = D^* \cap fD^*$. Let us write $hh''d' = fd_1 \ldots d_n$ with $d_i \in D (i=1,n)$. One of the fd_i belongs to $F = E - ED^+$, so h is either a prefix of a word of F or admits a word of F as a prefix. Therefore C is exhaustive. As $F \subset E \subset D^*$ we have $C^* \subset D^*$ so D is a pattern of C and it is clear that D is a pattern of the subtree below f since all the words of this subtree belong to D^*. Therefore D is overlapping.

iii $\rightarrow$ i) If D is overlapping, there exists a code C and a word $f \in C^l$, $f \notin D^+$ such that D is a pattern of C and a pattern of the subtree below f. As f is not the empty word and as we suppose that the cardinality of the alphabet is strictly superior to one we may find at least one word $c \in C$ which does not admit f as a prefix. Let us branch a subtree identical to B at the end of C. This construction gives us a tree A for which D is not an admissible pattern.

The following proposition generalizes a result of Perrin [6] from the regular case to the general case:

Proposition II.2. Let D be an overlapping code, then $C^* = \{f \in X^* \mid \forall d \in D^* \; \exists d' \in D^* \text{ such that } fdd' \in D^*\}$ is generated by an admissible pattern C of D.

We will now state a last theorem whose demonstration will appear elsewhere [3] and which is a generalization of a result of Perrin [8]. This theorem shows that the notions of systematic prefix code and principal pattern play an important role in the structure of regular codes. In the statement of the theorem A and B will be two regular codes, U and V will be two codes, U_1 and U_2 (V_1 and V_2) will be two disjoint subsets of U(V), U'_1 and U'_2 will be two disjoint subsets of a code U'.

Theorem III.8. $\mathcal{M}(A^*)$ and $\mathcal{M}(B^*)$ are isomorphic if and only if the three following conditions are satisfied

i) $\exists U = U_1 + U_2$ and $\exists V = V_1 + V_2$
such that
$A = U_2 + U_1V_2^*V_1 \qquad B = V_2 + V_1U_2^*U_1$

ii) $U_2^*U_1(V_2^*V_1)$ is not a systematic prefix code, or if it is a systematic prefix code then U(V) is its principal pattern.

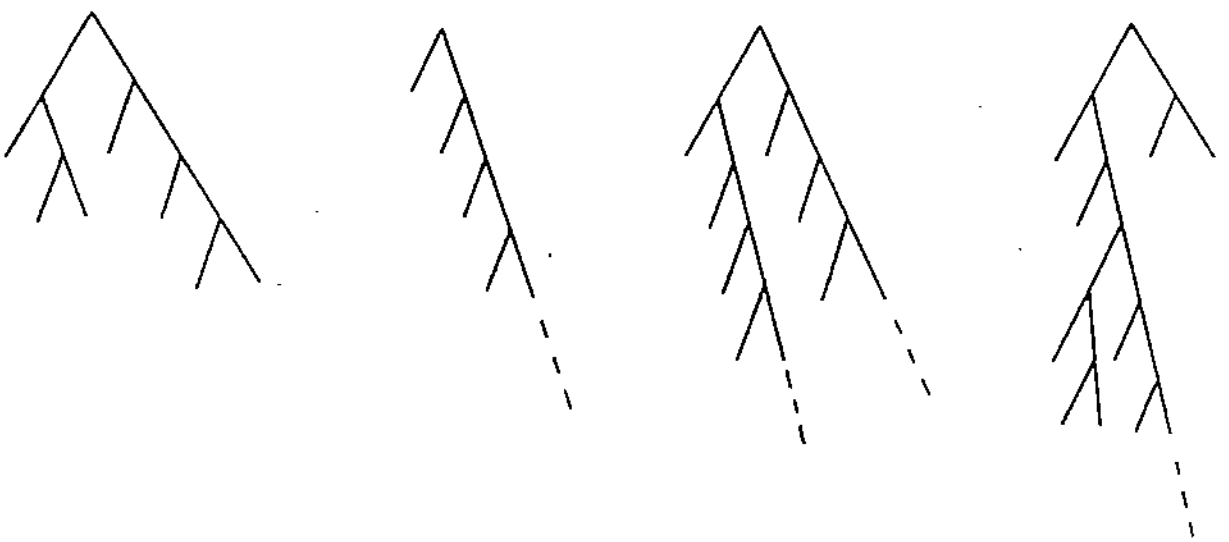
iii) If $A = U_2 + U_1V_2^*V_1 = U'_2 + U'_1V_2^*V_1$ then $U_2 \subset U'_2$ and the dual proposition for B.

Example of basis:

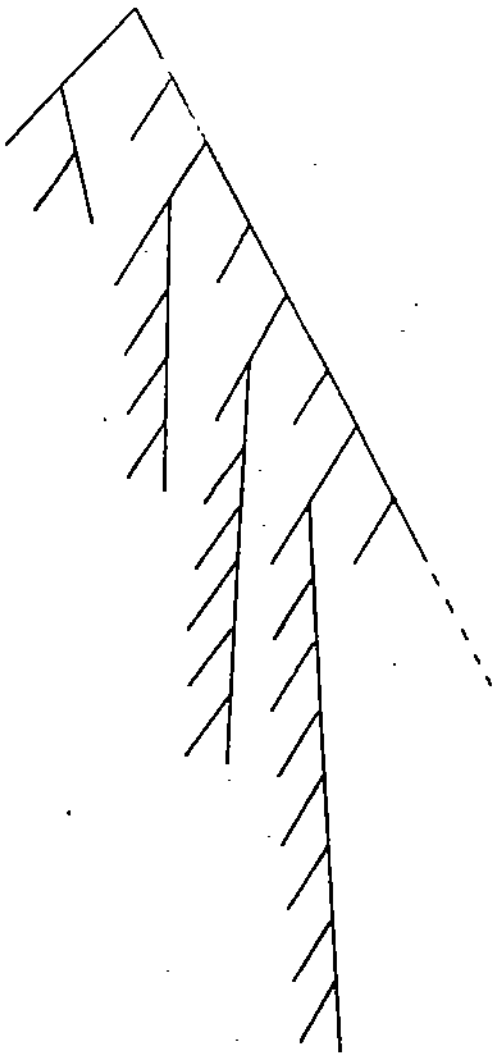
$C = x + yx + yy$ $\qquad A = (x+yy)^{*}yx$

principal pattern of A.

Various bases of A:



A non regular basis:



BIBLIOGRAPHY

1. A. Ginzburg. Algebraic theory of Automata, Academic Press, 1968.

2. J. L. Lassez. On the Structure of Systematic prefix codes. International Journal of Computer Mathematics. Vol. 3, Sept. 72.

3. J. L. Lassez. Prefix codes and Isomorphic Automata. International Journal of Computer Mathematics. To appear.

4. M. Nivat. Elements de la Theorie Generale des Codes in Automata Theory. Caianello Ed. Academic Press, 1966.

5. P. G. Neumann. Efficient error limiting variable length codes. I.R.E. Trans. Inf. Theory, 118, July, 1962.

6. D. Perrin. Le Langage Engendre Par un Code Prefixe et son Monoide Syntaxique. These, 1969. Paris Institute de Programmation.

7. D. Perrin. Sous-Monoides et Automates, Seminaire Schutzenberger-Lentin-Nivat. Problemes Mathematiques de la Theorie des Automates. Annee, 1969-170. Institut Menri Poincare.

8. D. Perrin. Codes Conjugues. Information and Control, Vo. 20, No. 3, April, 1972.

9. D. Perrin and J. F. Perrot. Sur les Codes Prefixes Complets Finis. C. R. Ac.Sc. Paris, 269, (1969).

10. J. F. Perrot. On the relationship between finite automata, finite monoids and prefix codes. Proceedings of the ACM Symposium on Theory of Computing. (1970).

11. J. F. Perrot. Endliche Automaten und Prefix Codes in "Automaten Theorie und Formale Spracken" (J. Dorr and G. Hotz, Eds). Tagungsbericht, Oberwolfach. (1969)

12. M. P. Schutzenberger. On an application of semigroup methods to some problems of Coding. IRE Trans. on Inf. Th., 172, 1956.